
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

FSI311

Bridgewater Associates: Inspectable models at scale

Lore Anaya Pozo (they/them)
Architect
Bridgewater Associates

Robin Anil (he/him)
Director, Research Technology
Bridgewater Associates

Anthony Pasquariello (he/him)
Senior Solutions Architect
AWS

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- Bridgewater’s inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

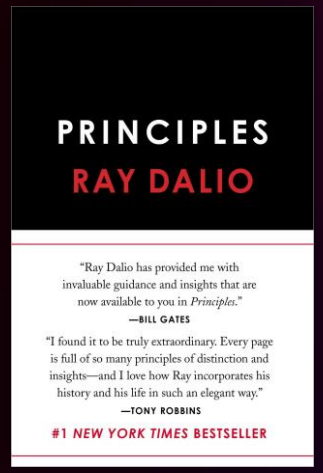
Inspectable models at scale

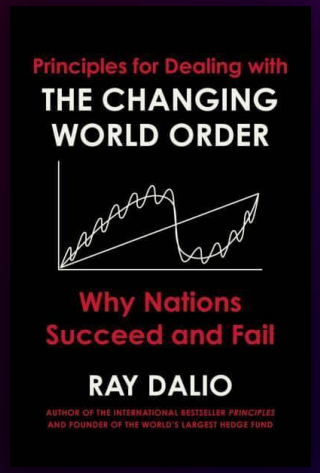
- Bridgewater's inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

About Bridgewater

- Founded by Ray Dalio
- Global macroeconomic investment manager
- We endlessly pursue the right answer to how economies work
- Our economic model is complex, constantly evolving for over 45 years

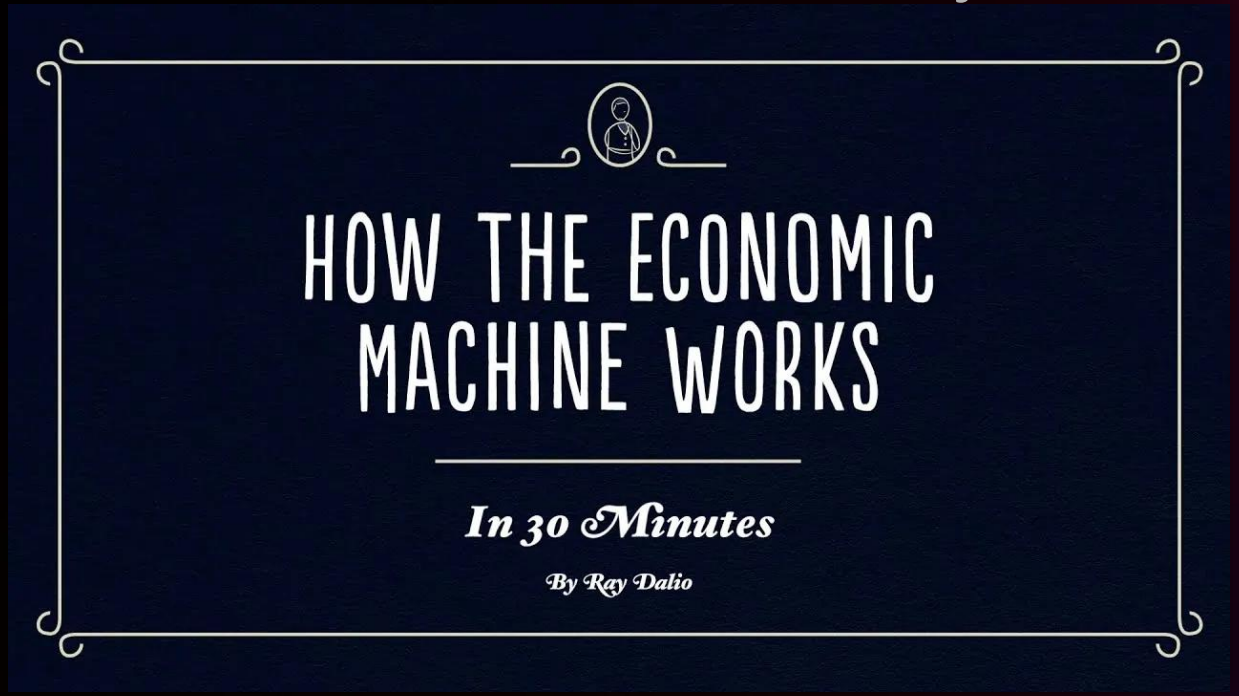




aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Bridgewater's model of the economy

- Research timeless and universal economic and investment principles
- We take these principles and write them as code
- We run this code, our economic model, continuously



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Bridgewater's inspectable model of the economy

- Good models are inspectable and explainable
- Why does our model show X and not Y?
- Why were we wrong? Why were we right?
- How do different conditions affect model behavior?
- How do we evaluate changes to our model?

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Economic models must be forward-looking

An economic model needs to have a good answer to data never seen before

- Unit tests and back-testing prevent regressions
- More than back-testing we need forward-testing: validation as new data flow in
- Every day is a new test of the model

The model cannot rot, that requires continuous oversight

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Our compute architecture runs on AWS

- Bridgewater's inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

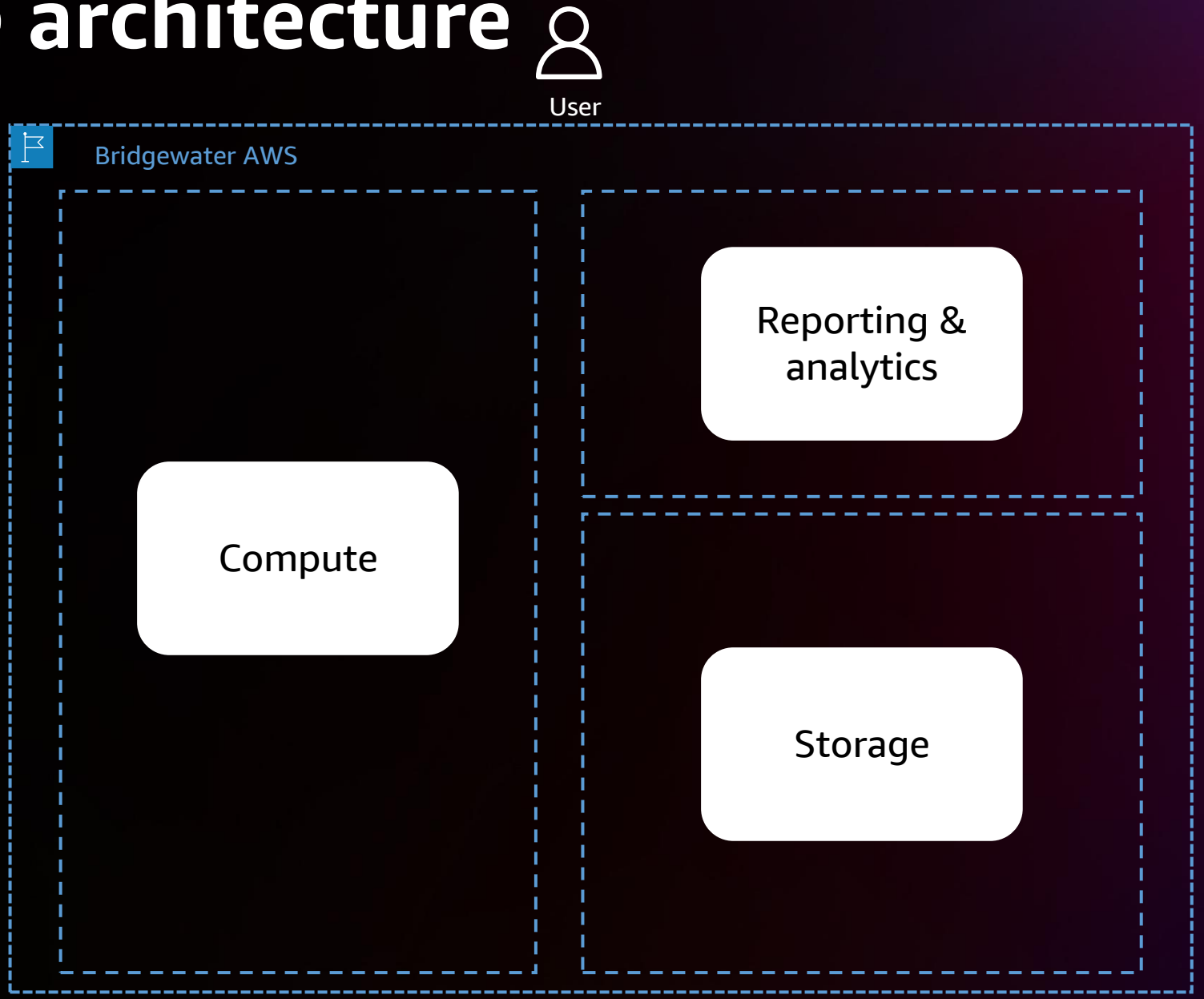
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Bridgewater's economic model runs on AWS

- New data flows in
- Forward-testing every day
- Back-testing all model changes
- Research workflows with hypothetical model changes (what-if)
- Consistently producing trades in all the markets we operate
- AWS based architecture enabled us to reliably scale out

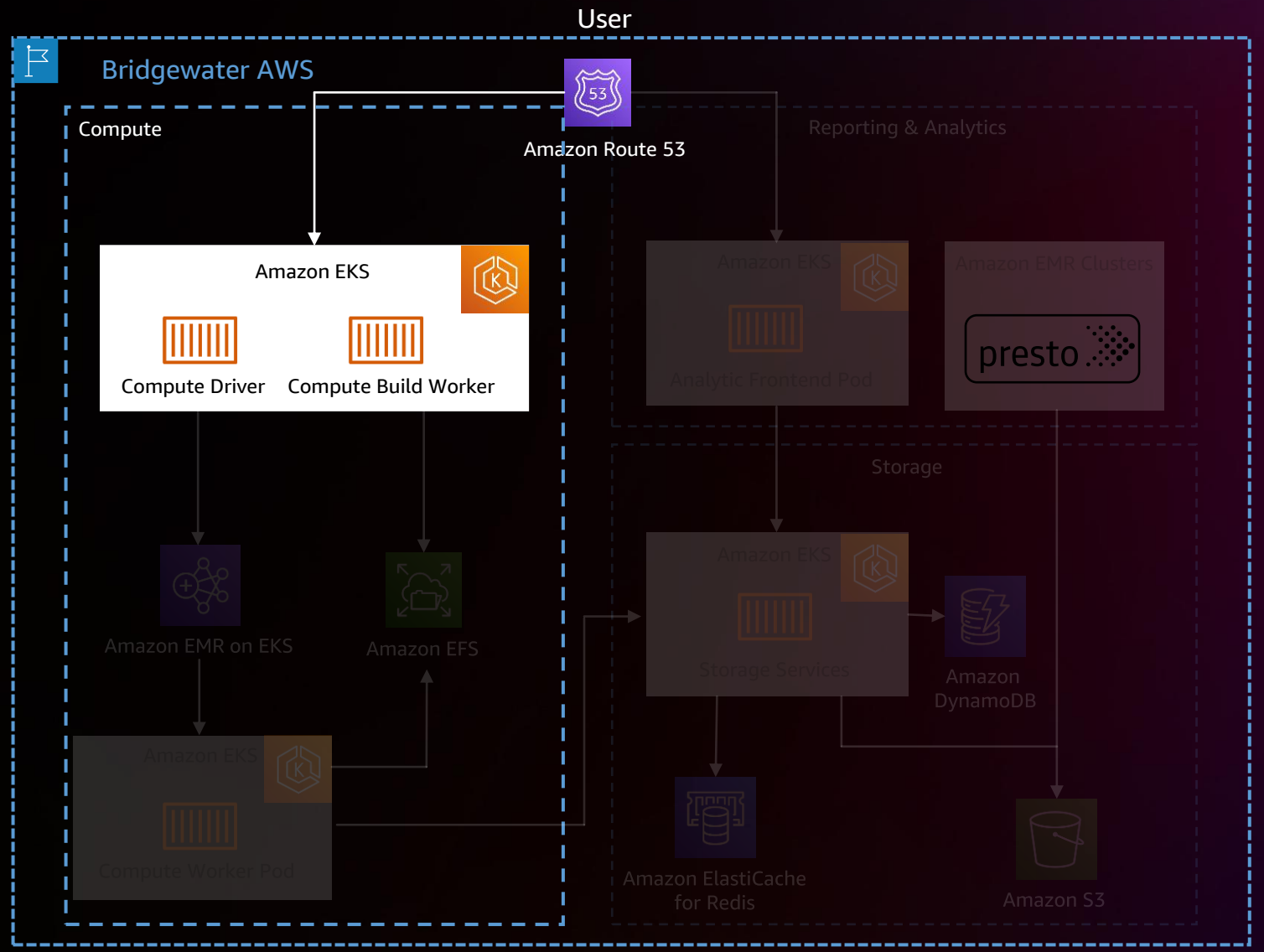
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



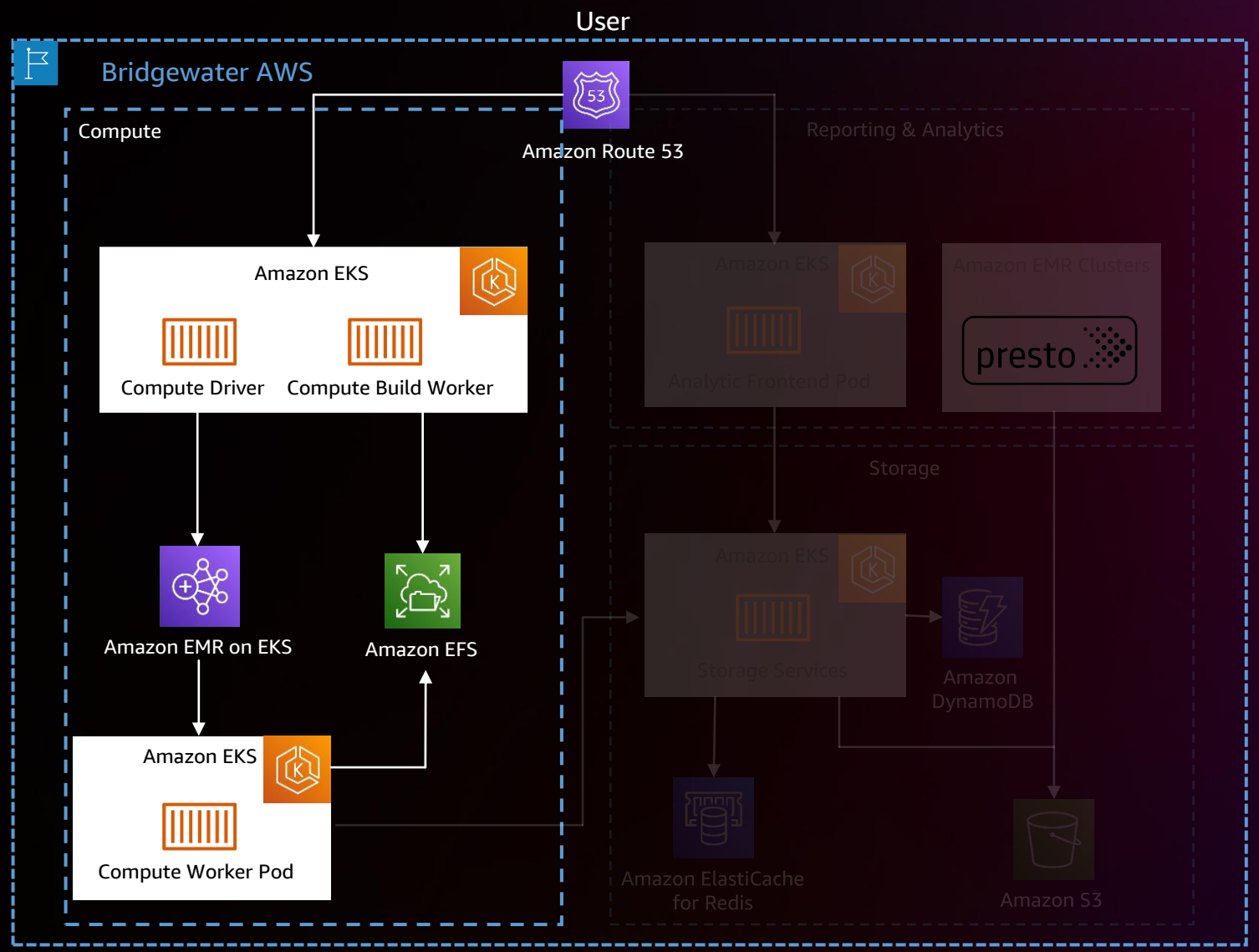
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



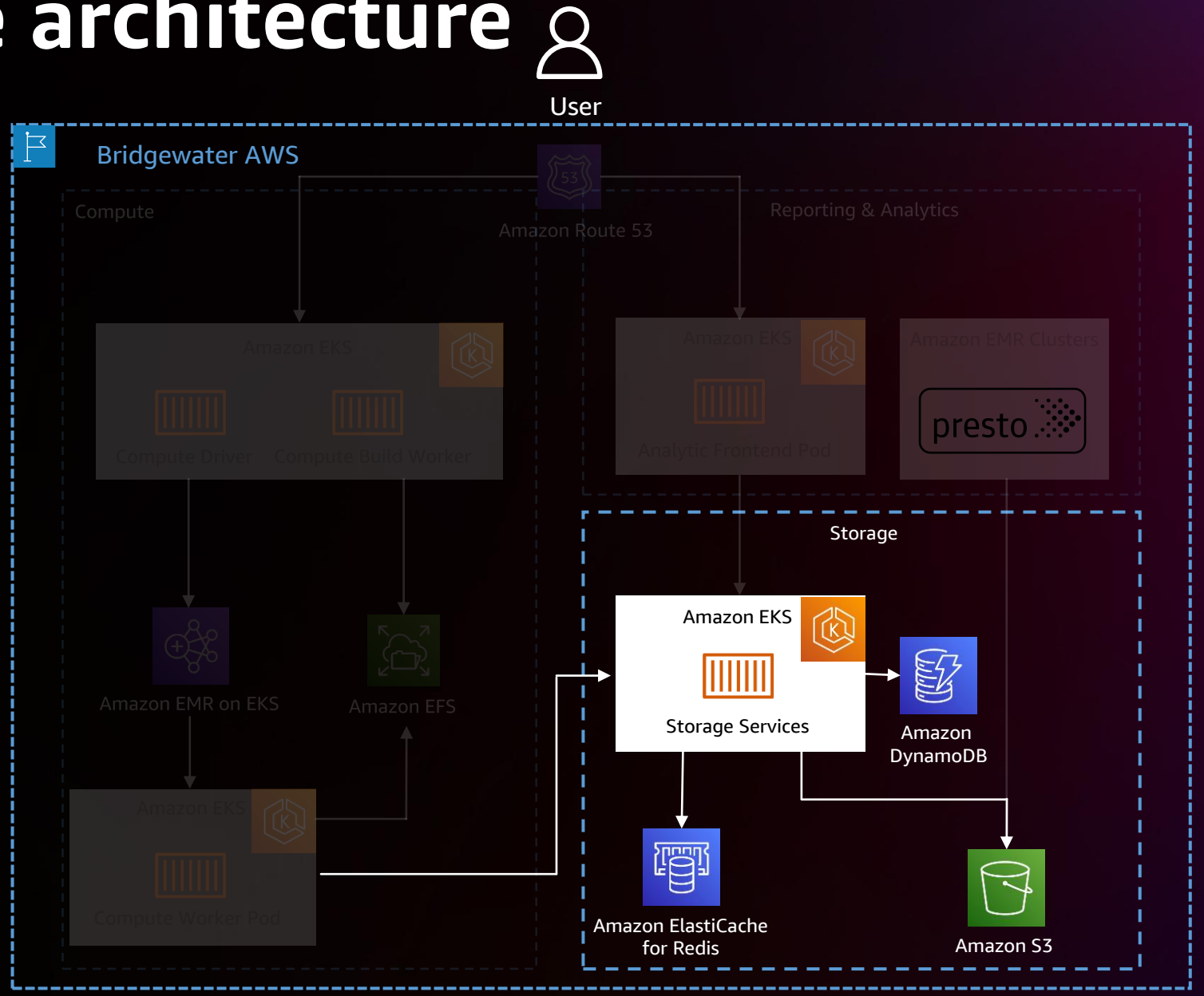
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



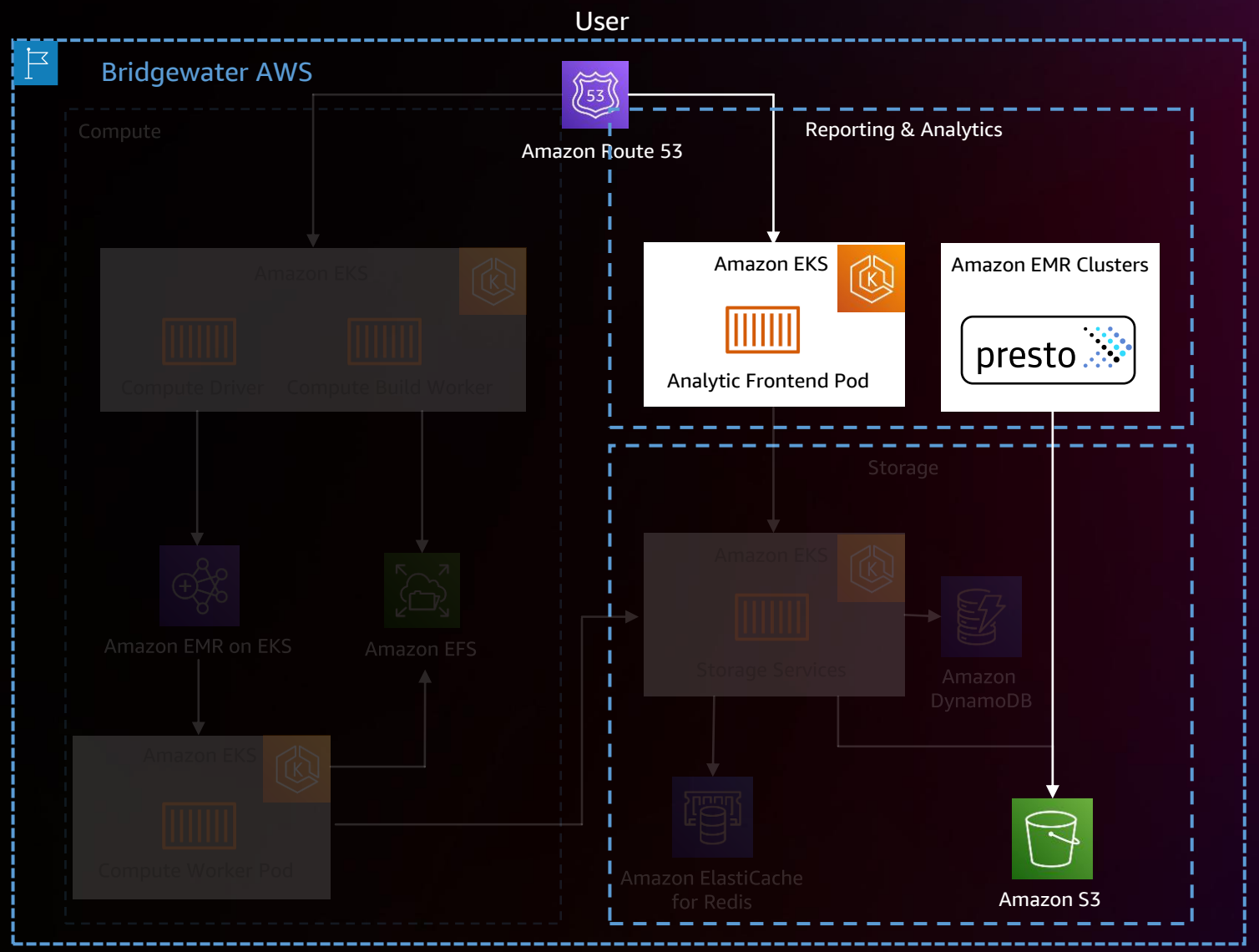
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



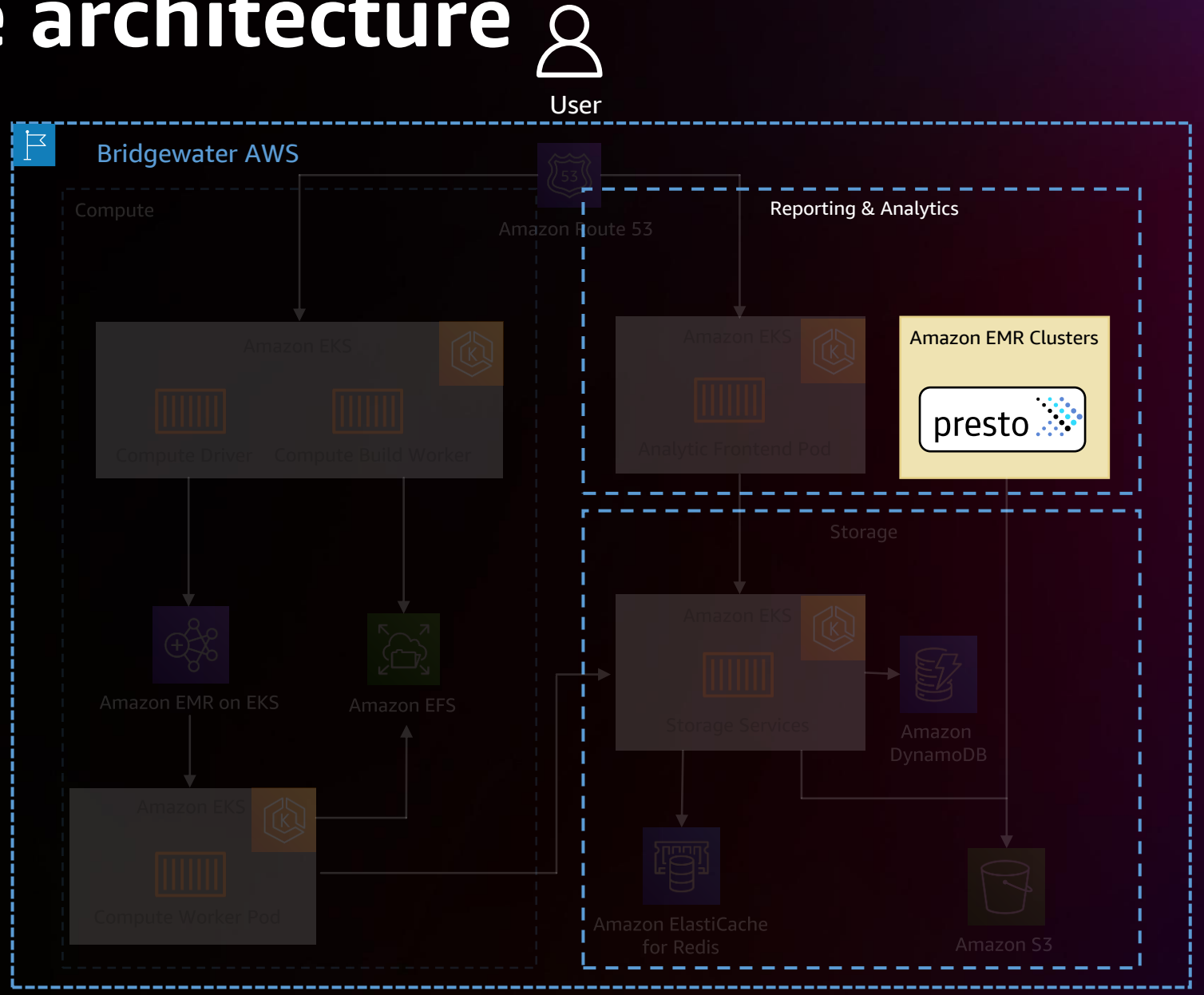
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



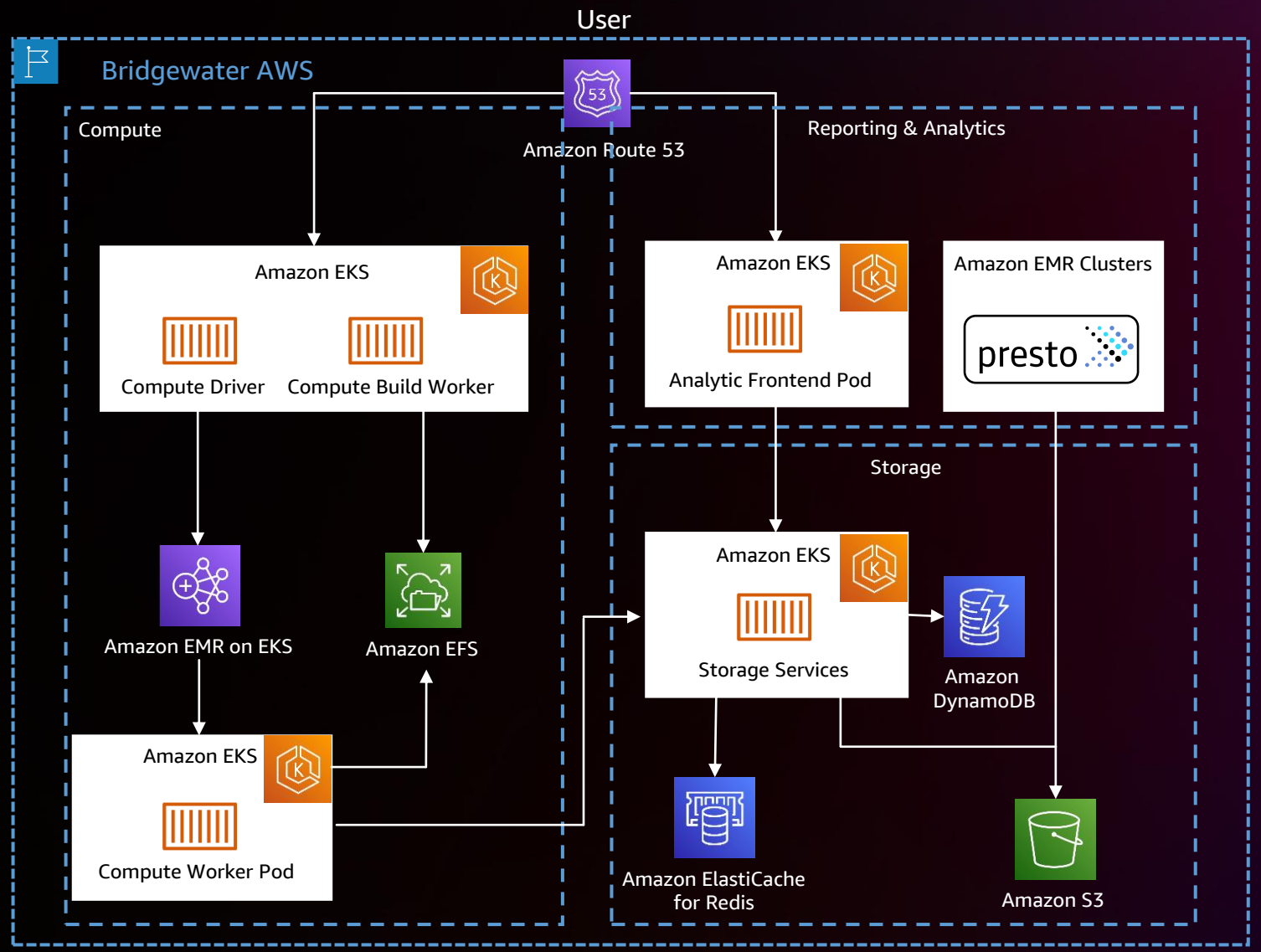
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Compute architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Bridgewater Research by the numbers

20+
petabytes

120 B+
time series

10K+
daily mini-models

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How we provided an inspectable model

- Bridgewater's inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How do we practically forward-test?

Return to the problem: how to continuously validate model behavior

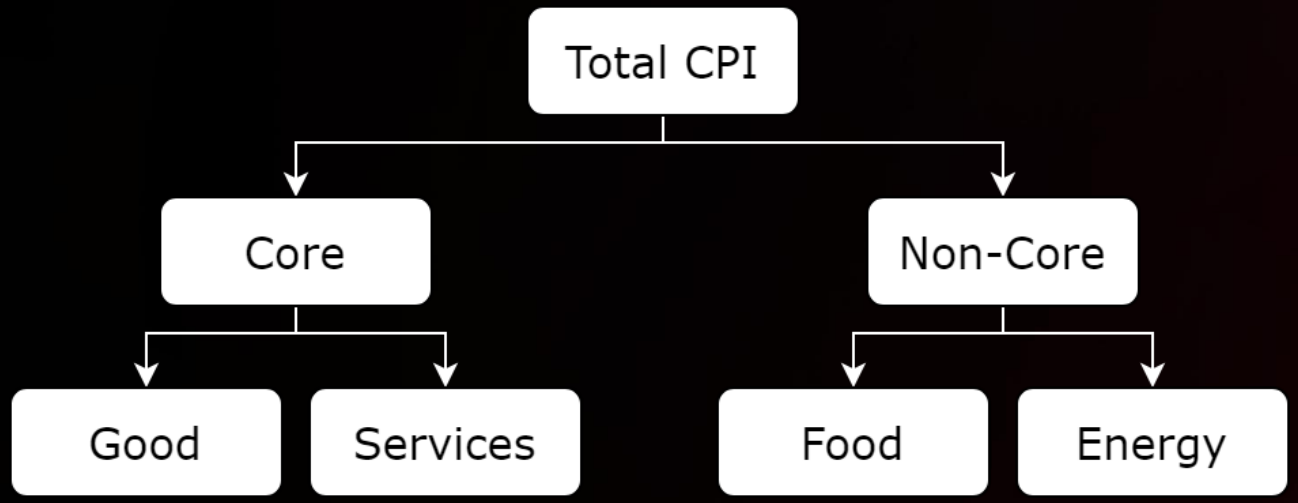
- The model is complex, a person cannot retain it in memory
- Data sizes are well into billions, people cannot sift in bounded time
- Issues are to be expected and plentiful – abnormal things happen
  - Someone must diagnose each issue: gap? bug? upstream data issue?
  - We cannot fire up a debugger to step through the code for every issue
  - We cannot stop trading every time we find an issue
  - Operationally, we needed to make our model inspectable!

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Example mini-model: Inflation

Change in the consumer price index (CPI) is a common measure for inflation




```
TotalCPI = Core + NonCore
Core     = Good + Services
NonCore  = Food + Energy
```

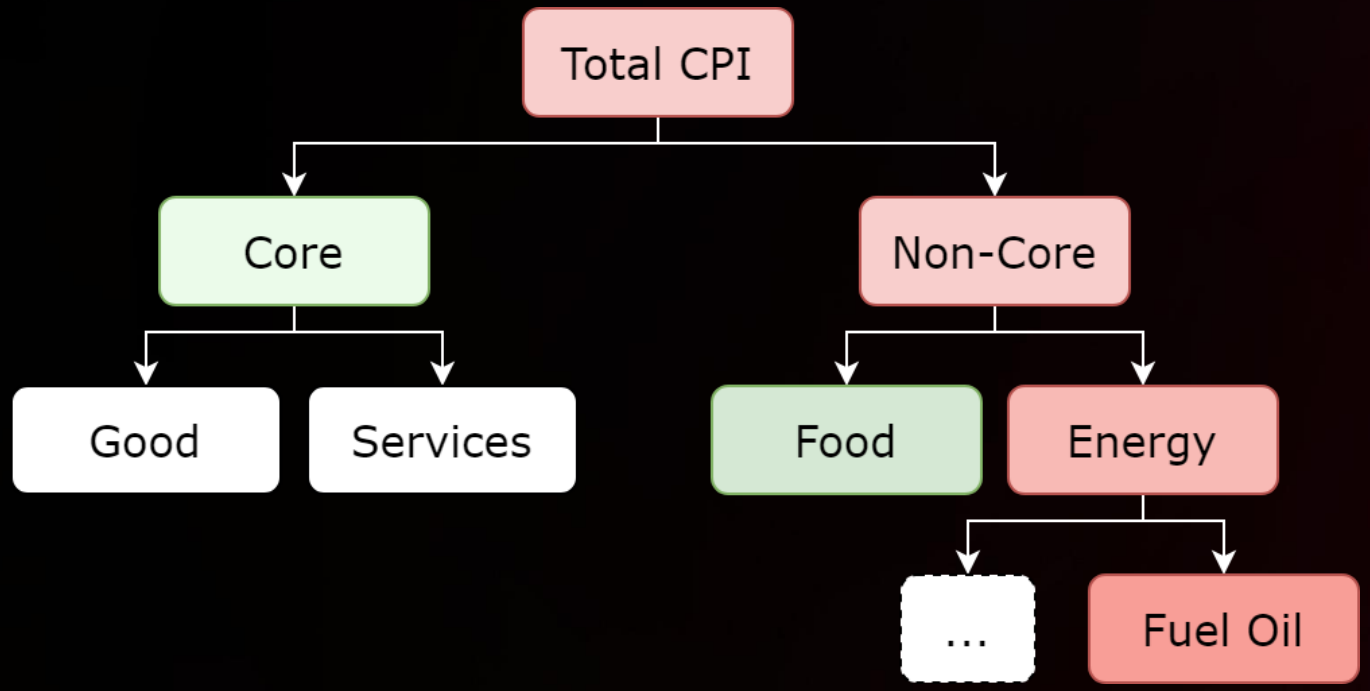
Query: What is driving the change in USA's Total CPI?

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Example mini-model: Inflation

Query: What is driving the change in USA's Total CPI?

Inspectable mini-model with provenance




```
TotalCPI = Core + NonCore
Core     = Good + Services
NonCore  = Food + Energy
…
```

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How we provided an inspectable model

- Inspectable models with our provenance system
- User has transparency into any mini-model
  - They can ask queries like “what is driving this change?”
  - They are productive without holding the system in their head
- Platform persists the intermediates and traces
- Users ask questions in an app that provides answers using mini-model’s results, intermediates, and traces
- Users loved it! Huge win for investment research and operations teams

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Where that solution fell short

- Bridgewater's inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Where our solution fell short

- We run 10,000s of mini-models daily
  - Producing hundreds of terabytes of new data
  - Each mini-models produces up to millions of time series, billions of points
  - Now we can confidently operate all those mini-models
- We start picking up more mini-models – operations can handle it
- Adoption goes up! Our economic model gets more sophisticated
- New research projects unlocked
- But what will give?

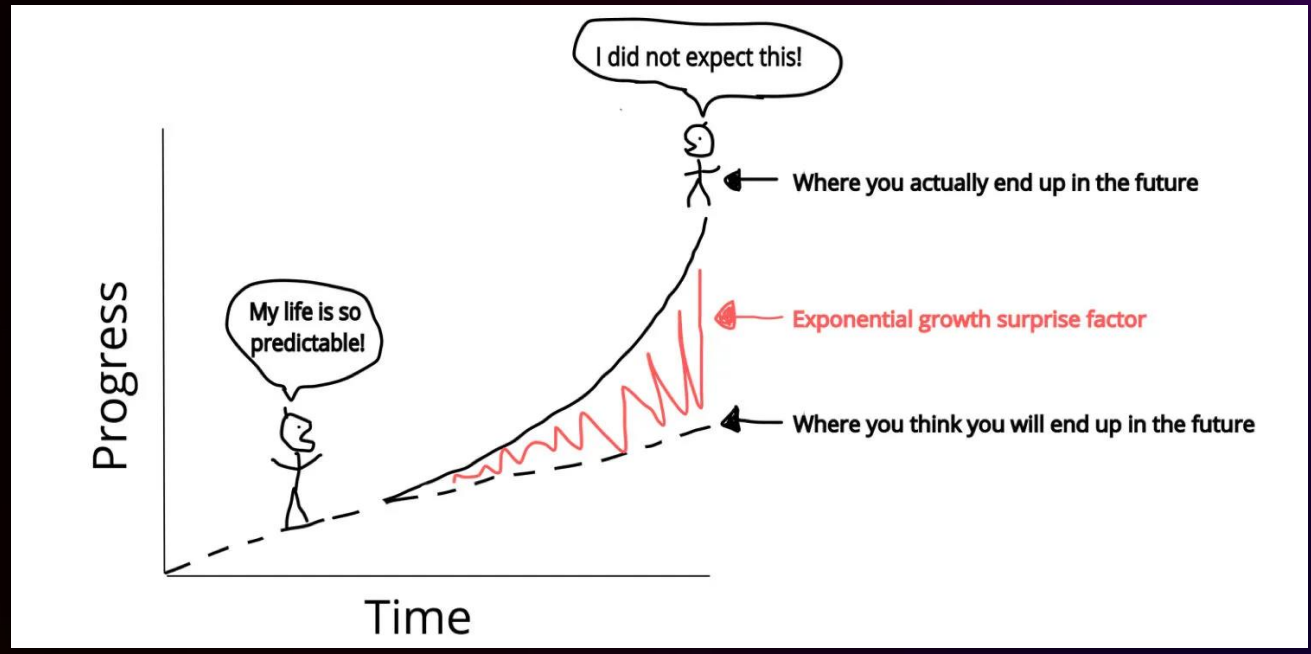
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Where our solution fell short

New mini-models, more complex, orders of magnitude larger data sizes…

- Provenance traces causes mini-models to run slower
- Data-intensive research projects inhibited
- Forward looking storage and compute doomed to skyrocket



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Where our solution fell short

We were spiraling into a negative cycle

- Users were unhappy with performance ☹
- Tech hit with “scale” related incidents ☹
- Leadership saw increased AWS cost ☹
- Storing logic traces was not going to work ☹

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How we REALLY SOLVED inspectabilty

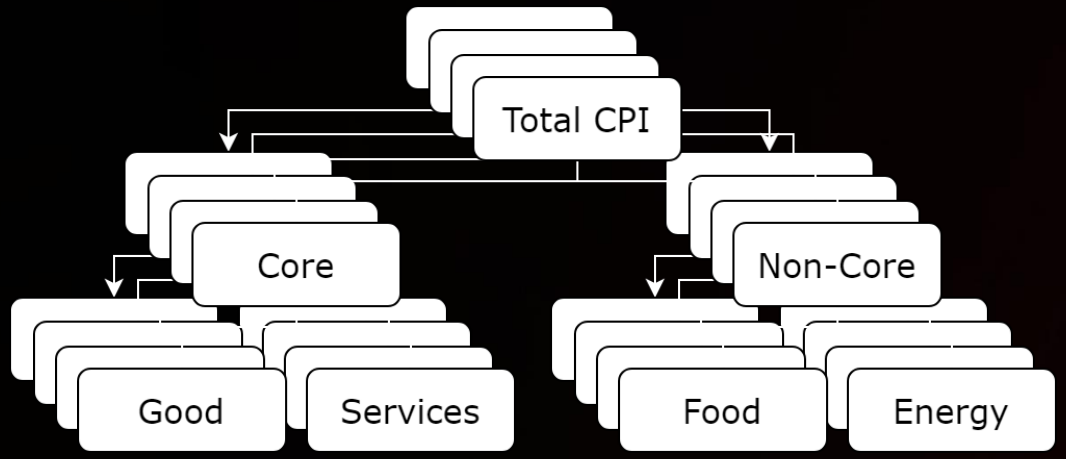
- Bridgewater's inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Reducing provenance burden on producer

Provenance was coupled with model runtime

- The runtime and storage burden scaled with data sizes and logic complexity
- That's roughly O(data × logic)
- Like running all mini-models in debug mode and dumping stacks and data

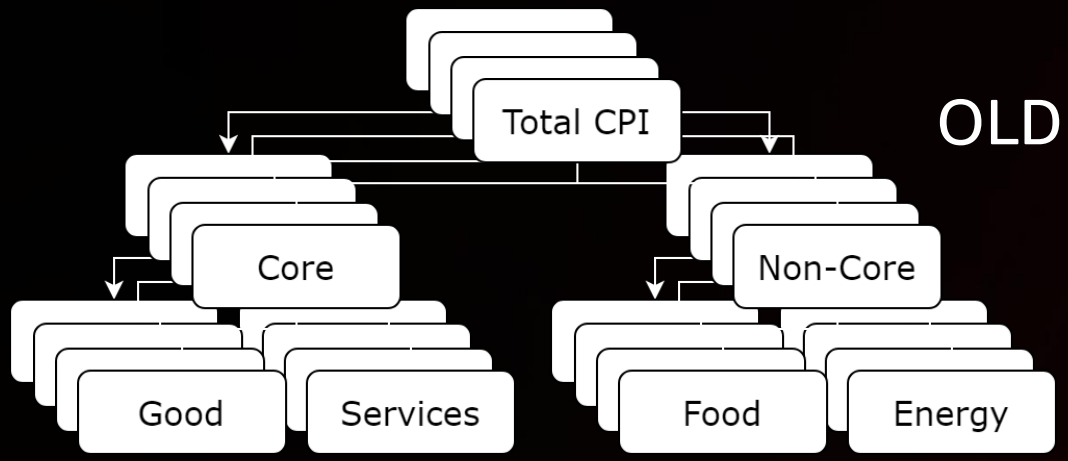


aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Reducing provenance burden on producer

Use static analysis to decouple provenance with model runtime

- Burden becomes O(data + logic)
- Persist the model's logic: it's a better encoding than the exploded traces



OLD ⟶ NEW

```
∀ country : Countries
TotalCPI = Core + NonCore
Core     = Good + Services
NonCore  = Food + Energy
```

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Consumer-side of provenance at scale

User query: What is driving the change in USA's Total CPI?

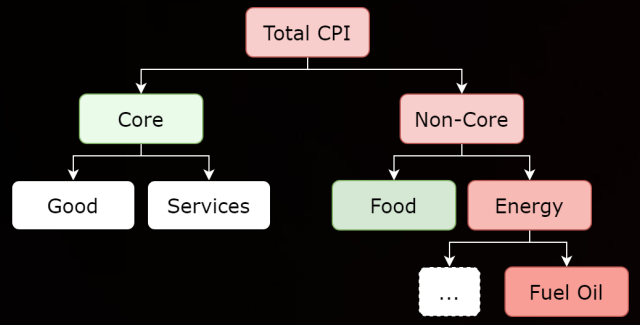
+ static provenance

```
∀ country : Countries
TotalCPI = Core + NonCore
Core     = Good + Services
NonCore  = Food + Energy
```

→ SQL

```
SELECT r.TotalCPI, i.Core, i.NonCore
FROM Results r
INNER JOIN Intermediates i ON r.country = i.country
WHERE country = "USA"
```

→ App



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Requirements for query engine

- Cannot regress performance serving interactive human-facing web interface
- Must support UDFs, which could operate on our time-series data
- Must read ORC tables in Amazon S3
- Should be SQL compliant
- Should have flexibility for caching solutions we may need for harder queries
- Must be managed by AWS (Amazon EMR)

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Technologies considered for query engine

- Amazon Redshift
  - Initial load is too slow, requires model changes to optimize
- Amazon Athena
  - UDFs too slow for fast single time-series calculation due to network overhead
  - Cannot add local caching with HDFS
- Apache Spark
  - Fixed overhead setup cost, wouldn’t be snappy
- These solutions could be made to work but require large engineering effort

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Query engine decision: Presto

Landed on Presto on Amazon EMR

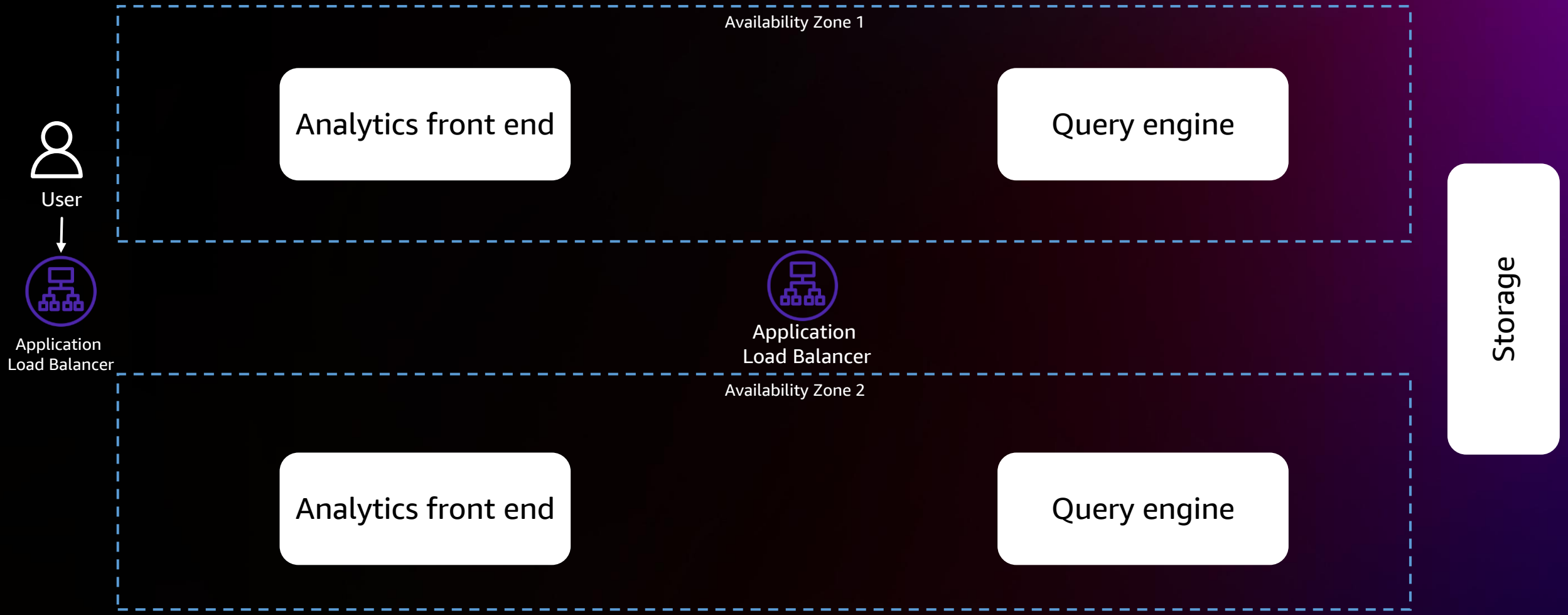
- UDFs are executed in-process
- Low overhead querying tables from Amazon S3
- Many seamless caching options

Additionally needed data catalog for Presto, that choice has its own pros/cons

- We started with AWS Glue for metadata

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Presto architecture



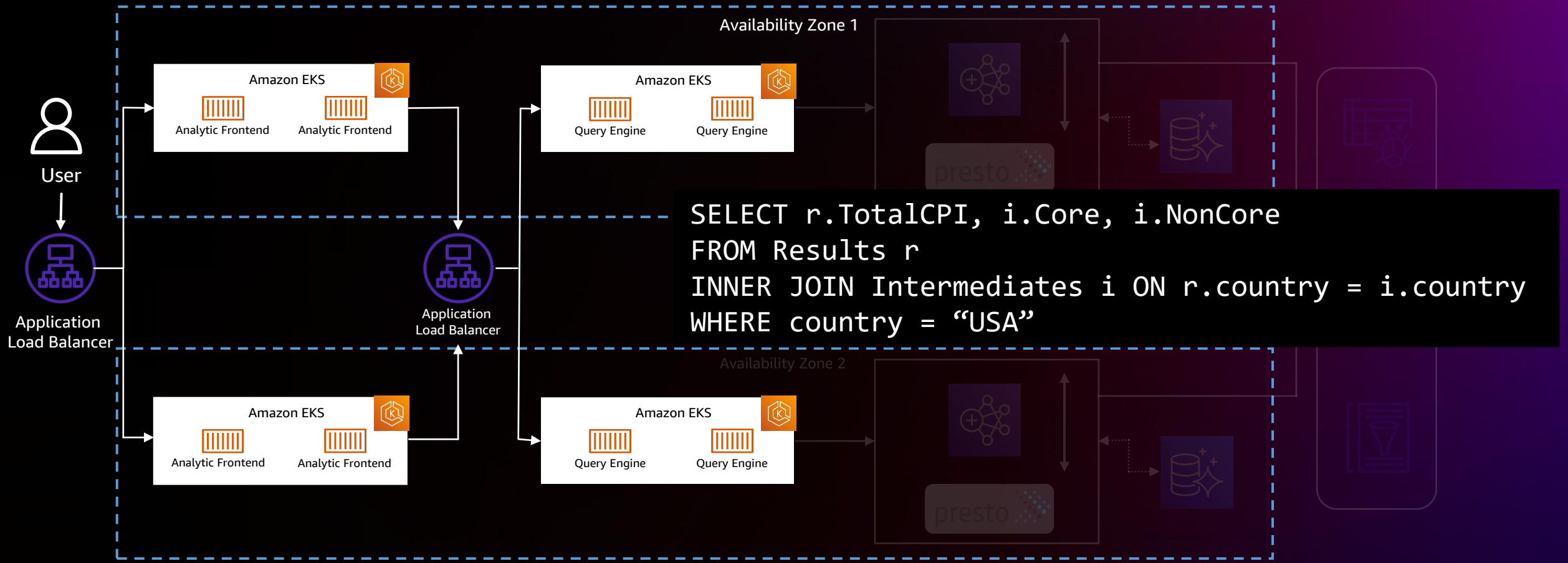
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Presto architecture



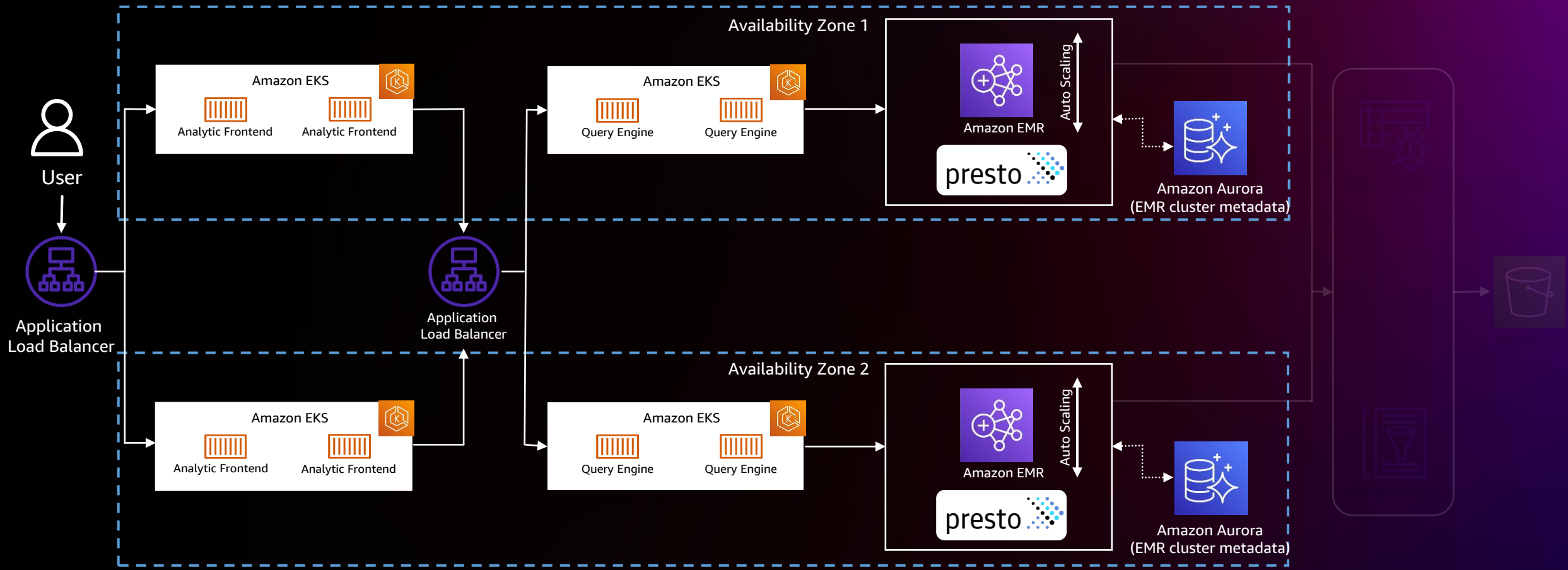
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Presto architecture



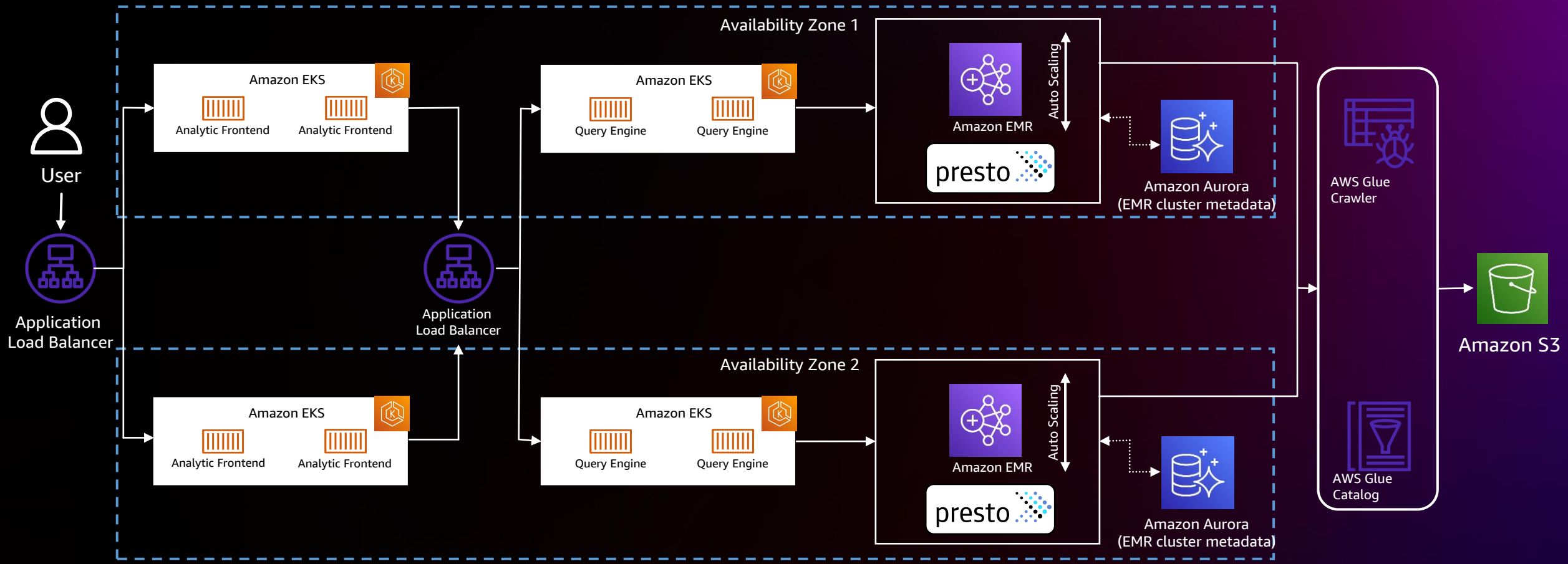
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Presto architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Presto architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon EMR Presto was not without its challenges

Rough start as Presto's default configuration was far from our needs

AWS Glue became our primary bottleneck

- Initially optimized with Amazon ElastiCache
- Later moved metastore to Amazon Aurora

Some interactions required <300ms interactivity

- We know these query patterns, dedicated a batch Presto cluster for pre-work
- Materialized "pre-work" tables were a lot faster to query using an interactive Presto cluster

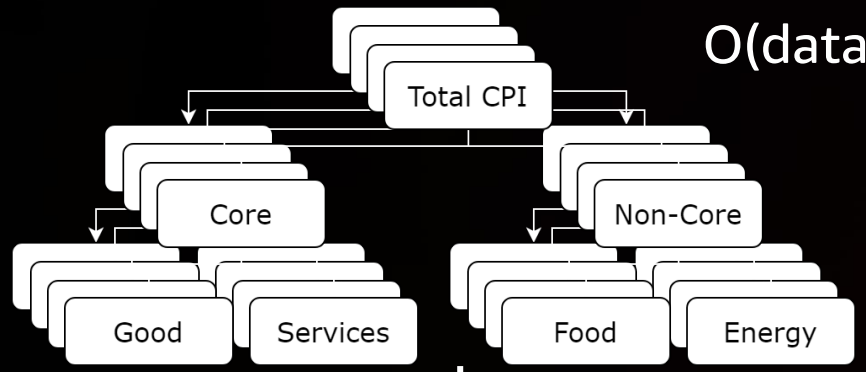
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Consumer-side of provenance at scale

User query: What is driving the change in USA's Total CPI?

OLD	NEW
O(data × logic)	O(data + logic)

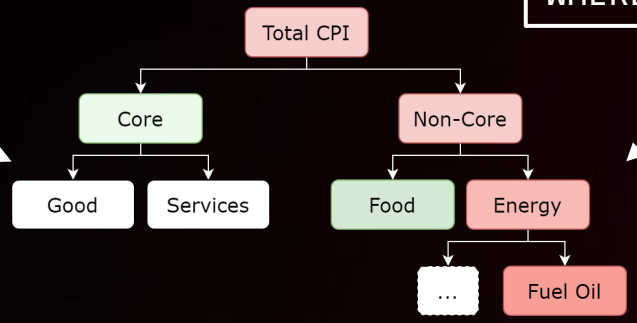


+ User Query

```
∀ country : Countries
TotalCPI = Core + NonCore
Core     = Good + Services
NonCore  = Food + Energy
```

+ User query

```
SELECT r.TotalCPI, i.Core, i.NonCore
FROM Results r
INNER JOIN Intermediates i ON r.country = i.country
WHERE country = "USA"
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

How we REALLY SOLVED inspectability

- Scale is proportional to what users inspect
  - Up to 4x runtime reduction for our biggest mini-model
  - Still snappy for inspection
- Decoupled from data size — for our needs, no longer a bottleneck
- We can optimize in ways previously not possible, across the architecture
- We can build new features previously not possible like sorting & limiting
- We now get linear scale for inspectable models with Presto on Amazon EMR
- Our proper cloud solution benefits from continued evolution of AWS

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Inspectable models at scale

Bridgewater is enabled with an inspectable model of the economy

The new architecture is a recipe for success:

- Easier to evolve the economic model, enabling new research
- Easier to evolve the internal platform, enabling new features and greater scale
- Pebble is in the right shoe: AWS provides scale while we focus on new business challenges

Performance improvements have no end! How do we get to next level of scale?

Product requests keep coming! We must build an architecture for the future

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Impact of our new architecture

- Bridgewater's inspectable model of the economy
- Our compute architecture
- How we provided an inspectable model
- Where that solution fell short
- How we REALLY SOLVED inspectability
- Impact of our new architecture

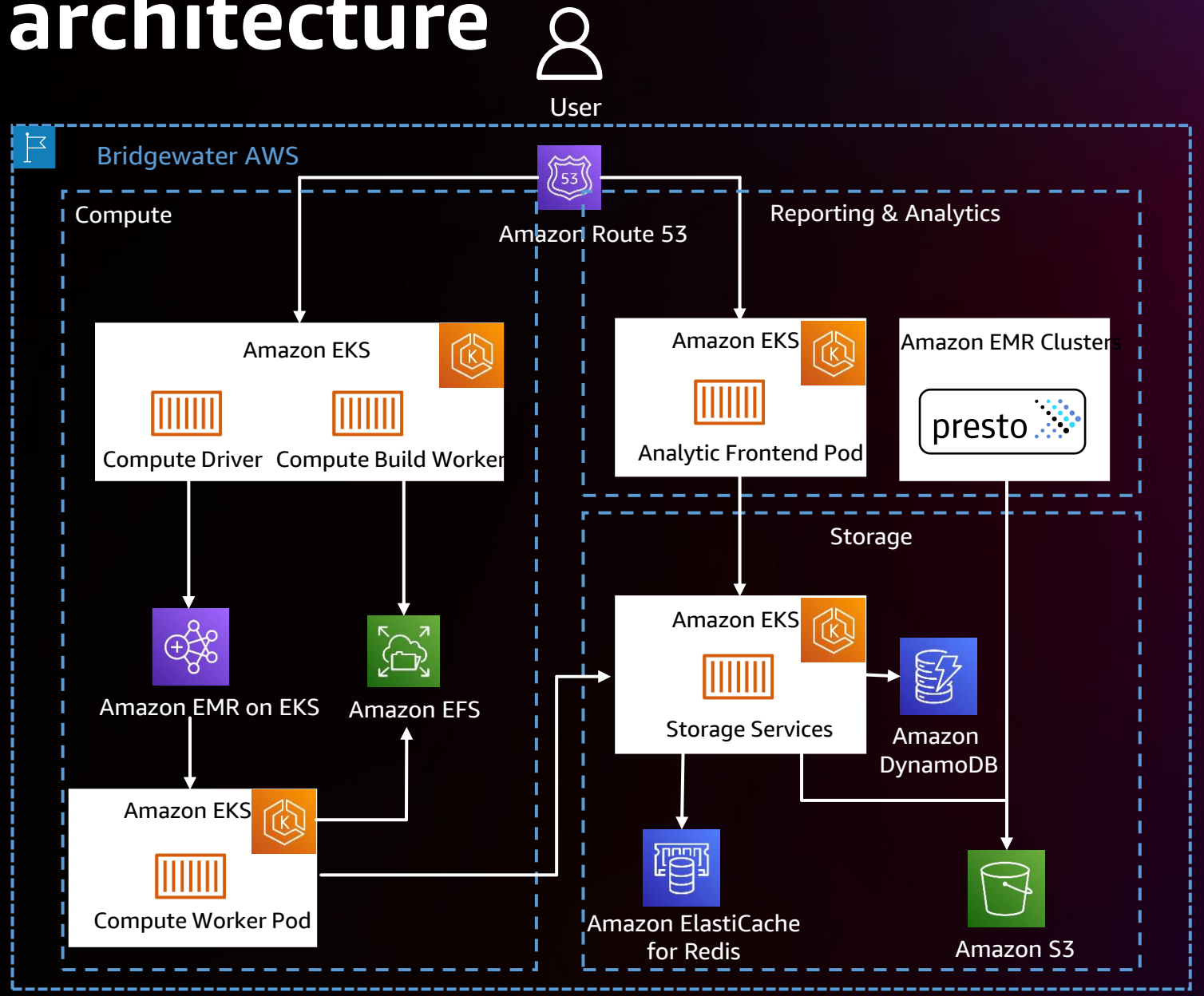
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Impact of our new architecture

- Runtime moves from producer to consumer at a fraction of cost
- Models are written by economists
  - Now provenance execution is owned by engineers, allowing for low-level optimization
- Fits neatly into our decoupled compute/storage model
- Enables better CPU utilization of mini-models
- Allows for larger models with linear runtime growth

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Our new architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Impact of our new architecture

2x

Reduction in avg runtime

75%+

Machine utilization

5x

Scale of simulations

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Looking back, we've come a long way

Built decoupled designs

- Simple to understand and easy to optimize

Managed infrastructure on AWS

- Avoided "Not invented here" — focus on what matters for us

Architected for bad days

- Turbo button to scale up after we have outages

80% reduction in total downtime hours in 2022!

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

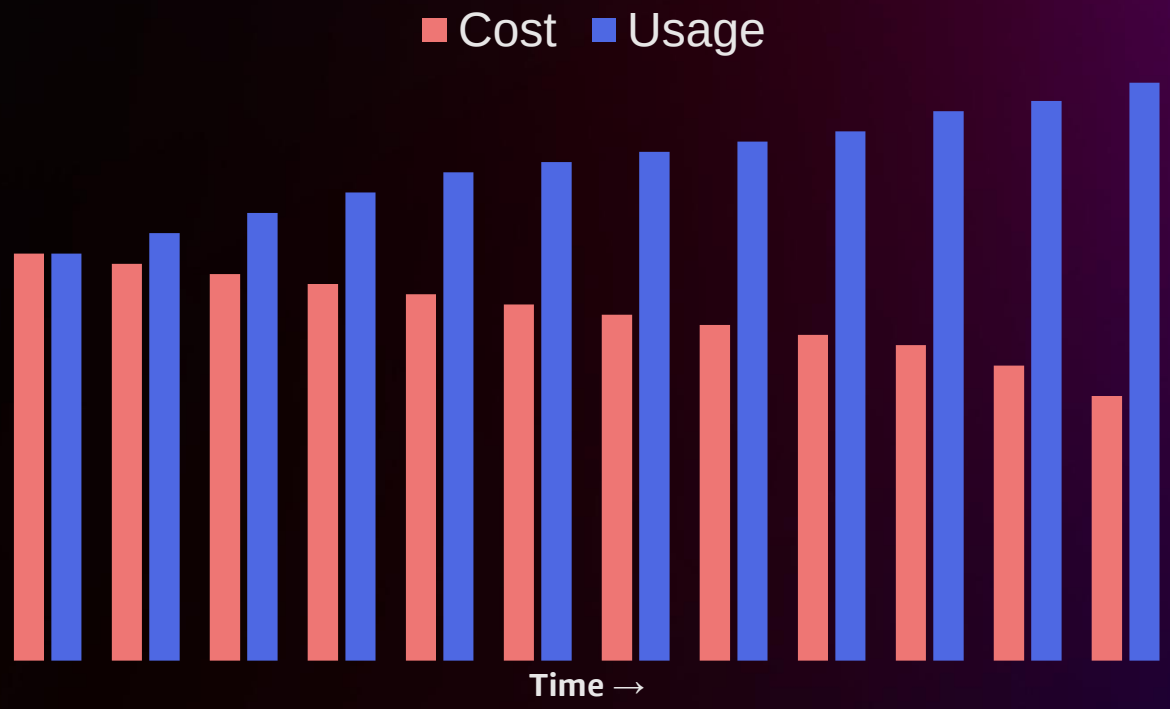
Saving cost while increasing usage

S3 Intelligent-Tiering

Monthly storage is **up 42%** compared to this time last year, while **spend is down 35%**

Amazon EMR

Running **20% more** simulations for the **same cost**



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lessons learned

Security	Decouple architecture to ease security design
Iterate	Frequent iteration tracks towards optimal solution
Long-term design	Work backwards from ideal
First principles	Understand and track core requirements

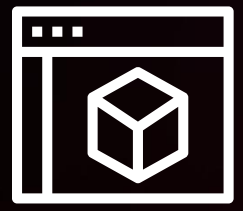
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Bridgewater & AWS relationship



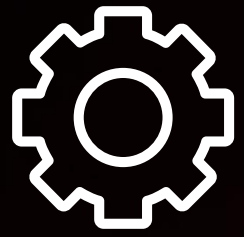
Roadmap planning



Architecture reviews



Executive alignment



AWS Data Lab



Optimization exercises

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!



Please complete the session survey in the **mobile app**

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.